

REMARKS

Applicant respectfully requests reconsideration and allowance of the subject application. Claims 1-40 are pending in this application.

In the December 6 Office Action, the Examiner requested that any response be accompanied by a 3½ inch IBM format floppy disk containing a duplicate copy of the response. In accordance with this request, such a floppy disk accompanies this response.

Applicant thanks the Examiner for the telephonic interview of October 4, 2001, at which the pending claims and the Matsumoto (U.S. Patent No. 5,835,765) and Kubo (U.S. Patent No. 5,881,284) references were discussed but no agreement on the allowability of any of the claims was reached.

Claims 1-39 stand rejected under 35 U.S.C. §103(a) as being unpatentable over U.S. Patent No. 5,835,765 to Matsumoto (hereinafter "Matsumoto") in view of U.S. Patent No. 5,881,284 to Kubo (hereinafter "Kubo"). Applicant respectfully submits that claims 1-39 are not obvious over Matsumoto in view of Kubo.

Matsumoto is directed to a computer operation management system for a computer operating system capable of simultaneously executing multiple application programs (see, Abstract). The underlying operating system in Matsumoto has a virtual storage area with a finite size – when a program is executed exceeding this finite size the program ends immediately (see, col. 9, lines 5-7). The management system of Matsumoto is situated between the operating system and the application programs, and operates to monitor and control execution of the application programs (see, Fig. 1 and col. 8, lines 50-60). Rather than having the operating system terminate an application, the management

1 system of Matsumoto monitors resource usage and initiates its own recovery  
2 process if a threshold is exceeded (see, col. 19, lines 52-57; and col. 16, lines 29-  
3 32).

4 The management system of Matsumoto includes a resource manager that  
5 checks an amount of memory used and then compares the actual memory used  
6 with a control limit read from a program definition file (see, col. 16, lines 29-32).  
7 If the control limit is exceeded, an error recovery processor is notified (see, col.  
8 16, lines 32-35). The error recovery process executes a predefined error recovery  
9 process also stored in the program definition file (see, col. 10, lines 37-40). The  
10 error recovery process comprises what measures to implement when an error  
11 occurs, the recovery procedure, alarm notification, mail, and what measures to  
12 implement for any related programs (see, col. 15, lines 10-14).

13 Kubo is directed to scheduling a job in a clustered computer system (see,  
14 Abstract). In Kubo, a job selector selects jobs for execution by one of multiple  
15 different clusters (see, col. 3, lines 8-18). A measurement mechanism exists in  
16 each cluster to measure the resource utilization in the cluster after the previous  
17 measurement (see, col. 3, lines 22-26). A request can be sent to the job selector to  
18 start scheduling a new job for the cluster at two different times: one is when the  
19 measurement is completed (see, Fig. 3), and the other is when a job is completed  
20 (see, Fig. 4). In the first situation, when the measurement is completed, if the  
21 resource utilization is low then the request controller requests the job selector to  
22 start a new job in the cluster; however, if the resource utilization is not low, then a  
23 new job is not requested (see, col. 3, lines 41-52). In the second situation, when  
24 execution of a job is completed, the request controller judges whether the resource  
25 utilization of the cluster that completes execution is high (see, col. 3, line 65 – col.

1 4, line 3). If the utilization in the cluster is not high, then a request is sent to the  
2 job selector to start scheduling a new job for the cluster; however, if the utilization  
3 in the cluster is high, then such job selection is not required (see, col. 4, lines 4-9).  
4 Thus, although Kubo teaches two different trigger points (utilization high and  
5 utilization low) for determining whether to request a job be scheduled, which of  
6 these trigger points is used is based on whether a resource utilization measurement  
7 was just completed or execution of a job was just completed.

8         Additionally, when execution of a job is completed, the request controller  
9 judges whether the resource utilization is high based on a first threshold value that  
10 is set to a value near a target resource utilization rate and a second threshold value  
11 is set to a value larger than the first threshold value (see, col. 4, lines 16-20). If the  
12 measured value exceeds the second threshold value, then the resource utilization is  
13 judged to be high, and if not, then the utilization is judged not to be high (see, col.  
14 4, lines 20-23).

15         To establish a *prima facie* case of obviousness, three basic criteria must be  
16 met. First, there must be some suggestion or motivation, either in the references  
17 themselves or in the knowledge generally available to one of ordinary skill in the  
18 art, to modify the reference or to combine reference teachings. Second, there must  
19 be a reasonable expectation of success. Finally, the prior art reference (or  
20 references when combined) must teach or suggest all the claim limitations. **The**  
21 **teaching or suggestion to make the claimed combination and the reasonable**  
22 **expectation of success must both be found in the prior art, and not based on**  
23 **applicant's disclosure.** See, emphasis added, MPEP § 2142.

24         Applicant respectfully submits that it would not have been obvious to one  
25 of ordinary skill in the art to combine Matsumoto and Kubo. As discussed above,

1 Matsumoto is directed to monitoring execution of application programs on a  
2 computer and invoking an error recovery process if an amount of memory used  
3 exceeds a control limit. Kubo, on the other hand, is directed to scheduling  
4 execution jobs across multiple different processor clusters. Applicant respectfully  
5 submits that there is no teaching or suggestion to combine the monitoring and  
6 invoking of an error recovery process of Matsumoto with the job scheduling of  
7 Kubo.

8 However, assuming for the sake of argument that the Matsumoto and Kubo  
9 references are combined, Applicant respectfully submits that the combination does  
10 not disclose or suggest a method of controlling memory usage as recited in  
11 claim 1. Claim 1 recites:

12       setting a plurality of memory thresholds; and  
13       the operating system wielding, at increasingly critical memory  
14       thresholds, correspondingly increasing control over said one or more  
      application programs to reduce memory usage.

15 Kubo simply discloses determining whether a cluster is ready to have a new job  
16 scheduled for it. Which of two different resource utilization levels are used to  
17 make the determination are based on the event that caused the determination to  
18 occur. One such event is the resource utilization measurement that occurs at  
19 predetermined times (see, col. 3, lines 22-26 and 41-52). The other such event is  
20 job execution completion (see, col. 3, line 65 – col. 4, line 3). Applicant  
21 respectfully submits that these two events are not increasingly critical memory  
22 thresholds. There is no discussion or suggestion in Kubo of an “increasingly  
23 critical” nature to these events – they are simply two different events that can be  
24 used to determine whether a cluster is ready to have a new job scheduled for it.  
25

1 Kubo also discloses that, in determining whether a cluster is ready to have a  
2 new job scheduled for it due to completing a previous job execution, the  
3 determining is based on whether its second threshold is exceeded (see, col. 4, lines  
4 20-23). This decision is based on the second threshold, not on the first threshold  
5 (the first threshold appears to simply be a reference point for determining what the  
6 second threshold should be). Whether the measured resource utilization exceeds  
7 or does not exceed the first threshold is irrelevant – Kubo discloses no comparison  
8 of any values to this first threshold in order to determine whether job scheduling is  
9 needed. Thus, there is no concept in Kubo of increasing control at increasingly  
10 critical thresholds as claimed in claim 1. There are only two possible results in  
11 Kubo when completing a previous job execution – a job is selected for execution  
12 by the cluster or a job is not selected for execution by the cluster. Which of these  
13 results occurs is based solely on the second threshold – there is nothing about  
14 increasing the control over the cluster for the different thresholds.

15 Thus, Applicant respectfully submits that Kubo does not disclose or suggest  
16 the operating system wielding, at increasingly critical memory thresholds,  
17 correspondingly increasing control over said one or more application programs to  
18 reduce memory usage as claimed in claim 1.

19 Furthermore, Applicant respectfully submits that Matsumoto does not cure  
20 this deficiency of Kubo. In Matsumoto, a computer operation management system  
21 including a computer resource manager exists between the operating system and  
22 the applications (see, Figs. 1 and 2). This allows the management system to  
23 monitor resource usage and initiate its own recovery process if the threshold is  
24 exceeded, rather than having the operating system immediately terminate the  
25 application and generate an error (see, col. 19, lines 52-57; and col. 16, lines 29-

1 32). Thus, notification of the error recovery means is the only action taken  
2 because the management system is able to intervene before the operating system  
3 would have to terminate (abnormally end) the application – the abnormal endings  
4 are thus prevented (see, col. 19, lines 52-57).

5 Thus, Applicant respectfully submits that Matsumoto does not disclose the  
6 operating system wielding, at increasingly critical memory thresholds,  
7 correspondingly increasing control over said one or more application programs to  
8 reduce memory usage as claimed in claim 1. Matsumoto's goal is to replace one  
9 level of control (termination) with another level of control (notice), not to have the  
10 two levels coexist. Furthermore, it is two different entities that are providing the  
11 level of control – the management system provides the notice level while the  
12 underlying operating system provides the termination level. Applicant  
13 respectfully submits that there is no disclosure or suggestion in Matsumoto of a  
14 single entity providing both of these levels, and thus no disclosure or suggestion of  
15 an operating system wielding increasing control as claimed in claim 1.

16 Thus, for at least these reasons, Applicant respectfully submits that claim 1  
17 is allowable over Matsumoto in view of Kubo.

18 With respect to claim 2, it was asserted in the June 6 Office Action that  
19 “limiting, closing, or terminating of a program are well known in the art  
20 (MPEP2144.03)” (see, ¶4, p. 3). Applicant respectfully traverses this rejection and  
21 submits that “communicating a request to at least one of the application programs  
22 for the at least one **application program to limit its use of memory**” as claimed  
23 in claim 2 is not well known in the art. Applicant respectfully requests that the  
24 rejection to claim 2 be withdrawn or evidence cited teaching “communicating a  
25

1 request to at least one of the application programs for the at least one application  
2 program to limit its use of memory” as claimed in claim 2.

3 With respect to claim 3, it was asserted in the June 6 Office Action that  
4 “limiting, closing, or terminating of a program are well known in the art  
5 (MPEP2144.03)” (see, ¶4, p. 3). Applicant respectfully traverses this rejection and  
6 submits that wielding increasing operating system control comprising “prompting  
7 a user to select at least one of the application programs and then the operating  
8 system requesting that the at least one selected application program close itself” as  
9 claimed in claim 3 is not well known in the art. Applicant respectfully requests  
10 that the rejection to claim 3 be withdrawn or evidence cited teaching this element.

11 With respect to claim 4, it was asserted in the June 6 Office Action that  
12 “limiting, closing, or terminating of a program are well known in the art  
13 (MPEP2144.03)” (see, ¶4, p. 3). Applicant respectfully traverses this rejection and  
14 submits that wielding increasing operating system control comprising “prompting  
15 a user to select at least one of the application programs and then terminating it  
16 without allowing its further execution” as claimed in claim 4 is not well known in  
17 the art. Applicant respectfully requests that the rejection to claim 4 be withdrawn  
18 or evidence cited teaching this element.

19 With respect to claim 5, it was asserted in the June 6 Office Action that  
20 “limiting, closing, or terminating of a program are well known in the art  
21 (MPEP2144.03)” (see, ¶4, p. 3). Applicant respectfully traverses this rejection and  
22 submits that wielding increasing operating system control comprising:

23 at a first memory threshold, requesting at least one of the  
24 application programs to limit its use of memory;

25 at a second memory threshold, requesting at least one of the  
application programs to close itself; and

1 at a third memory threshold, terminating at least one of the  
application programs without allowing its further execution.

2 as claimed in claim 5 is not well known in the art. Applicant respectfully requests  
3 that the rejection to claim 5 be withdrawn or evidence cited teaching these  
4 elements.

5 With respect to claim 6, it was asserted in the June 6 Office Action that  
6 “limiting, closing, or terminating of a program are well known in the art  
7 (MPEP2144.03)” (see, ¶4, p. 3). Applicant respectfully traverses this rejection and  
8 submits that wielding increasing operating system control comprising:

9 at a first memory threshold, requesting at least one of the  
10 application programs to limit its use of memory;

11 at a second memory threshold, prompting a user to select at  
least one of the application programs and then requesting it to close  
12 itself; and

13 at a third memory threshold, prompting the user to select at  
least one of the application programs and then terminating it without  
allowing its further execution.

14 as claimed in claim 6 is not well known in the art. Applicant respectfully requests  
15 that the rejection to claim 6 be withdrawn or evidence cited teaching these  
16 elements.

17 With respect to claims 7-9, claims 7-9 depend from claim 1 and Applicant  
18 respectfully submits that claims 7-9 are likewise allowable over the cited  
19 references for at least the reasons discussed above with reference to claim 1.

20 With respect to claims 10-33, claims 10-33 are rejected for reasoning  
21 analogous to claims 1-8. Thus, Applicant respectfully submits that claims 10-33  
22 are likewise allowable over the cited references for at least the reasons discussed  
23 above.



1 With respect to claims 34-35, claims 34-35 depend from claim 32 and  
2 Applicant respectfully submits that claims 34-35 are likewise allowable over the  
3 cited references for at least the reasons discussed above with reference to claim 32.

4 With respect to claims 36-39, claims 36-39 are rejected for reasoning  
5 analogous to claims 32-35. Thus, Applicant respectfully submits that claims 36-  
6 39 are likewise allowable over the cited references for at least the reasons  
7 discussed above.

8 Claim 40 stands rejected under 35 U.S.C. §103(a) as being unpatentable  
9 over U.S. Patent No. 5,815,702 to Kannan et al. (hereinafter "Kannan") in view of  
10 U.S. Patent No. 5,826,082 to Bishop et al. (hereinafter "Bishop"). Applicant  
11 respectfully submits that claim 40 is not obvious over Kannan in view of Bishop.

12 Kannan discloses a system in which an exception handler intercepts  
13 exceptions that are fatal and that would normally cause an application to be  
14 terminated (see, col. 4, lines 44-47). The exception handler notifies a crash guard  
15 process of the fatal exception, which in turn interacts with the user to determine  
16 whether to continue executing the application or to terminate it (see, col. 4, lines  
17 47-51). The crash guard process displays a warning dialog that allows the user to  
18 choose between continuing to work and terminating the application (see, Fig. 4,  
19 and col. 7, lines 34-47). Thus, the system of Kannan allows a user to continue  
20 using an application that has generated a fatal exception that would otherwise have  
21 caused the operating system to terminate execution of the application (see, col. 2,  
22 lines 39-43).

23 Bishop, on the other hand, discloses a computer system including a  
24 resource manager that is responsible for allocation of the computer system's  
25 resources (see, col. 2, lines 35-37). A thread that needs a resource submits a

1 request to the resource manager for the necessary resource including a requested  
2 amount of the resource (see, col. 3, lines 53-62). If the requested amount is not  
3 available, then the resource manager searches for a prior request(s) of a separate  
4 thread that has already reserved enough of the resource to satisfy the request, and  
5 attempts to suspend the prior request(s) (see, col. 4, lines 52-57 and 63-67; and  
6 col. 5, lines 1-4 and 23-31). This suspending of the prior request is transparent to  
7 the user of the computer system (see, col. 5, lines 30-31).

8 Applicant respectfully submits that it would not have been obvious to one  
9 of ordinary skill in the art to combine Kannan and Bishop. As discussed above,  
10 Kannan is directed to allowing a user to continue using an application that has  
11 generated a fatal exception that would otherwise have caused the operating system  
12 to terminate execution of the application, whereas Bishop is directed to reserving  
13 resources and suspending prior requests transparently to the user. Applicant  
14 respectfully submits that there is no teaching or suggestion to combine the  
15 allowing a user to select between continuing and terminating an application that  
16 has generated a fatal exception of Kannan with the resource management and user-  
17 transparent resource request suspension of Bishop.

18 However, assuming for the sake of argument that the Kannan and Bishop  
19 references are combined, Applicant respectfully submits that the combination does  
20 not disclose or suggest an application program that resides in a computer-readable  
21 memory for execution by a processor as recited in claim 40. Claim 40 recites:

22 An application program that resides in a computer-readable memory  
23 for execution by a processor in conjunction with an operating  
24 system, the application program having a message loop that receives  
25 messages from an operating system, the application program being  
responsive to a particular message received through its message loop  
to reduce its current use of memory.

1 Kannan is not cited as disclosing, and Applicant respectfully submits that  
2 Kannan does not disclose or suggest, an application program being responsive to a  
3 particular message received through its message loop to reduce its current use of  
4 memory as claimed in claim 40.

5 Bishop, on the other hand, is cited as disclosing an application operation  
6 that is programmed to reduce its current use of memory (see, ¶5, p. 5). Applicant  
7 respectfully disagrees with this assertion. Bishop discloses a resource manager  
8 that is responsible for allocation of the computer's resources and that has the  
9 ability to suspend requests for resources (see, col. 5, lines 26-31). The resource  
10 manager determines whether to suspend a prior request of a thread based on the  
11 priorities of the current and prior requests, or the priorities of the current and prior  
12 requesting threads (see, col. 5, lines 17-22). Neither the current nor the prior  
13 requesting thread determines whether its request is to be suspended – the resource  
14 manager makes that determination. Thus, since neither of the threads makes this  
15 determination, neither of the threads itself is programmed (nor is the program the  
16 threads are part of programmed) to reduce its current use of memory (the resource  
17 manager forces one of their requests to be suspended, and thus forces a suspension  
18 in the use of resources by the thread whose request is suspended).

19 The cited portion of Bishop (col. 3, line 63-col. 4, line 5) discusses the  
20 beginning of the steps involved in reserving a resource. The cited portion of  
21 Bishop discloses determining if the requested amount of the requested resource is  
22 available (or if the amount available to be reserved is below a predetermined  
23 threshold). This determination is the basis for determining whether the resource  
24 manager needs to attempt to suspend a prior request (see, col. 4, lines 52-57).  
25 Applicant respectfully submits that nothing in this cited portion discloses or

1 suggests an application program being responsive to a particular message received  
2 through its message loop to reduce its current use of memory as claimed in claim  
3 40.

4 Thus, Applicant respectfully submits that Bishop does not cure the  
5 deficiencies of Kannan.

6 For at least these reasons, Applicant respectfully submits that claim 40 is  
7 allowable over Kannan in view of Bishop.

8  
9 **Conclusion**

10 Claims 1-40 are in condition for allowance. Applicant respectfully requests  
11 reconsideration and issuance of the subject application. Should any matter in this  
12 case remain unresolved, the undersigned attorney respectfully requests a telephone  
13 conference with the Examiner to resolve any such outstanding matter.

14  
15  
16 Date: December 6, 2001

Respectfully Submitted,

17 By: 

Allan T. Sponseller  
18 Reg. No. 38,318  
19 (509) 324-9256  
20  
21  
22  
23  
24  
25